Safety Critical Machine Learning

From Anomaly Detection to Computer Vision

2025

@simplymath

social.cryptography.dog

github.com/simplymathematics

But are they safe?

Is 99% accuracy good enough?

or even 99.999999% ?

Safety

SIL	On-demand Operation	Continuous Operation
I	$[10^{-6}, 10^{-5})$	$[10^{-10}, 10^{-9})$
II	$[10^{-7}, 10^{-6})$	$[10^{-11}, 10^{-10})$
III	$[10^{-8}, 10^{-7})$	$[10^{-12}, 10^{-11})$
IV	$[10^{-9}, 10^{-8})$	$[10^{-13}, 10^{-12})$

Table: Acceptable Failure Rates for different Safety Integrity Levels (SIL) in which a single death is possible, measured in failures per second (1).

Introduction to Adversarial Attacks

However, much research has highlighted the efficacy of attacks that aim to subvert these models.

- **Evasion Attacks:** deceive the model into misclassifying certain inputs (2).
- **Poisoning Attacks:** injects malicious dataset to influence the model's behaviour at run-time (3).
- **Inference Attacks:** obtain sensitive information about the training data or other confidential details (4).
- **Model Inversion Attacks:** infer sensitive information about the model by exploiting the model's outputs (5).

Introduction to Adversarial Attacks

However, much research has highlighted the efficacy of attacks that aim to subvert these models.

- **Evasion Attacks:** deceive the model into misclassifying certain inputs (2).
- **Poisoning Attacks:** injects malicious dataset to influence the model's behaviour at run-time (3).
- **Inference Attacks:** obtain sensitive information about the training data or other confidential details (4).
- **Model Inversion Attacks:** infer sensitive information about the model by exploiting the model's outputs (5).

Since we are primarily concerned with run-time failures, we focus on evasion attacks moving forward. These attacks a visualised inFigure 2

Attack Types Visualized

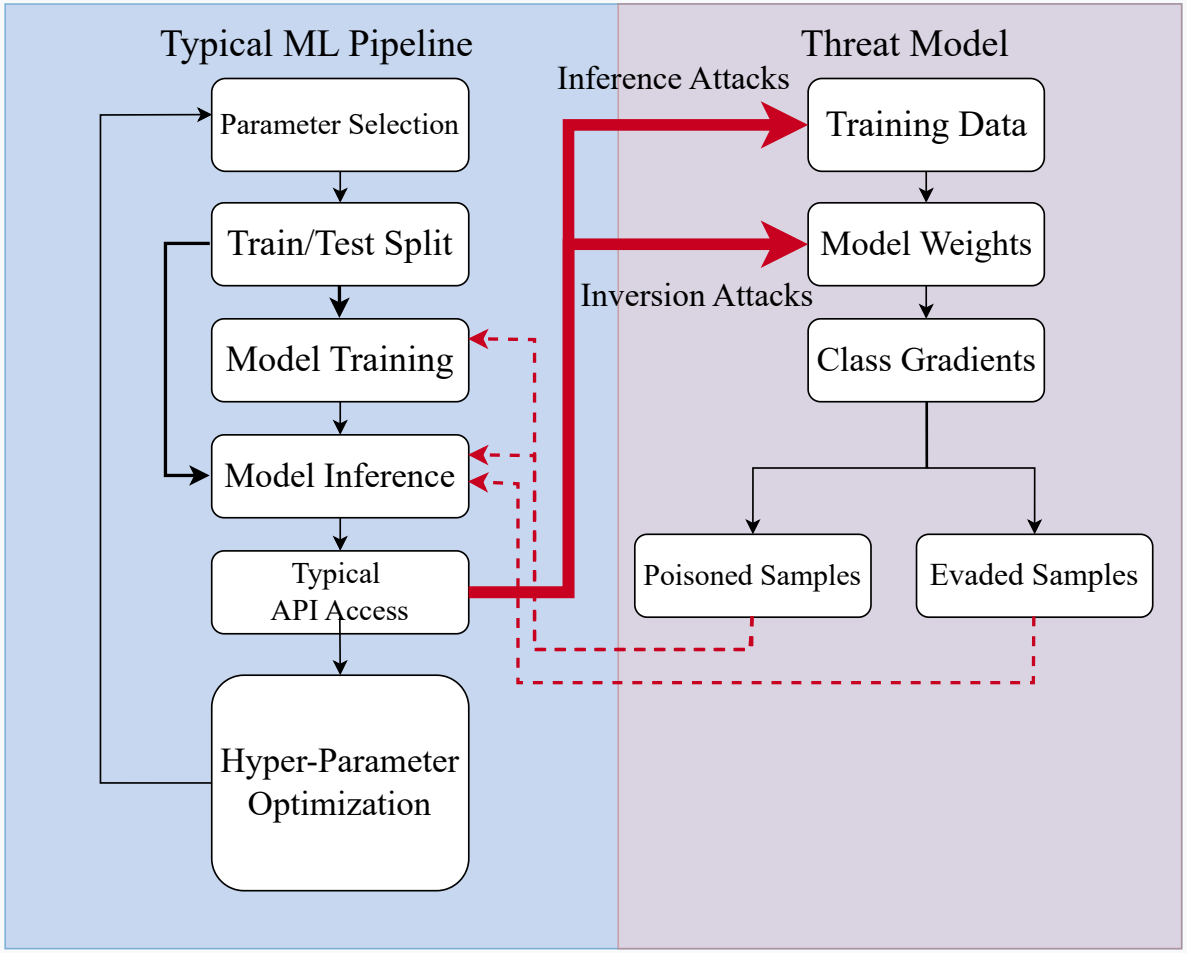


Figure: This diagram illustrates the taxonomy of machine learning attacks, as explained in the previous slide.

Adversarial Attacks Illustrated

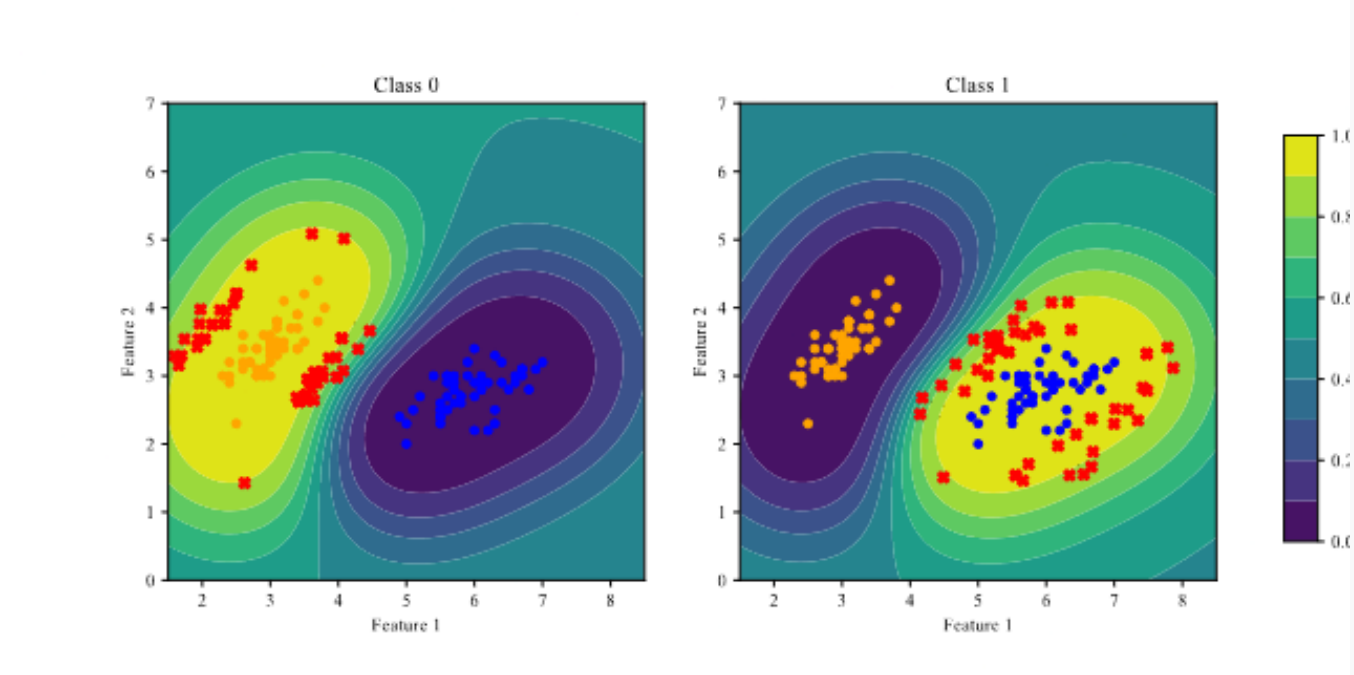


Figure: This figure depicts class 0 (orange) and class 1 (blue), as well as adversarial points for each class (red). The yellow-blue scale reflects the probability that a particular point belongs to class 1 such that 50% confidence. We can see that small perturbations in the feature space can create large deviations in the model classification.

Failure Rates and Robustness

$$\text{Accuracy} = 1 - \frac{\text{Incorrect Classifications}}{\text{Total Classifications}} \tag{1}$$

$$= 1 - \eta \tag{2}$$

where η is the probability of model failure.

$$\text{Failure Rate} = \frac{\text{False Classifications}}{\text{Total Time (s)}} t = \lambda(t) \tag{3}$$

$$\Delta\lambda = \frac{\lambda_{control} - \lambda}{\lambda} \tag{4}$$

Benign refers to the metric on normal test data and *adversarial* refers to the metric when tested against samples that include malicious noise.

Tested Defences

Gauss-in (ℓ_2): The 'Gaussian Augmentation' defence adds Gaussian noise to some proportion of the training samples.

Conf (ℓ_∞): The 'High Confidence Thresholding' defence only returns a classification when the specified confidence threshold is reached, resulting in a failed query if a classification is less certain.

Gauss-out (ℓ_2): The 'Gaussian Noise' defence, rather than adding noise to the input data, adds noise during inference (6)

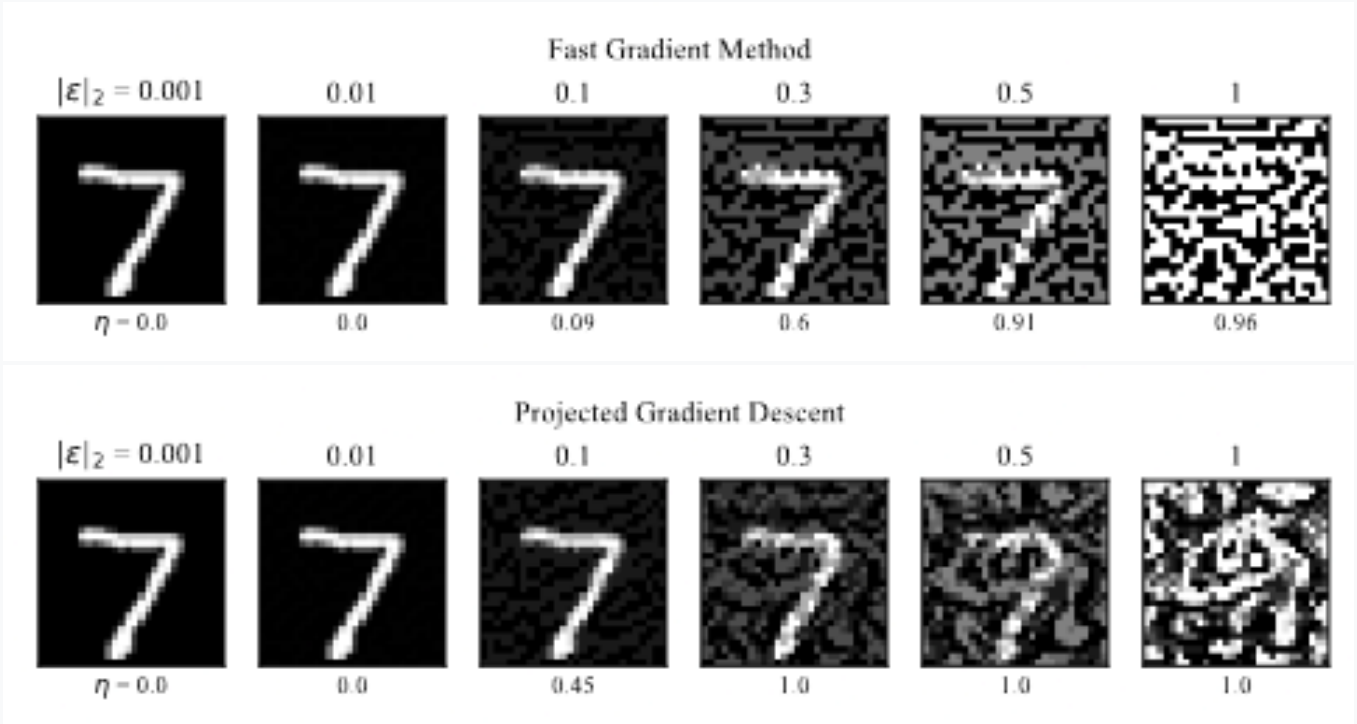
FSQ: The 'Feature Squeezing' defence changes the bit-depth of the input data to minimise the noise induced by floating-point operations.

Tested Attacks

FGM ($\ell_1, \ell_2, \ell_\infty$): The 'Fast Gradient Method' quickly generates a noisy sample, with no feasibility conditions beyond a specified step size.

PGD ($\ell_1, \ell_2, \ell_\infty$): The 'Projected Gradient Method' extends the FGM attack to include a projection on the ε-sphere, ensuring that generated samples do not fall outside of the feasible space (7).

Evasion Attacks Visualized



This figure depicts images as we increase the attack strength (ϵ), as well as the adversarial success rate, η.

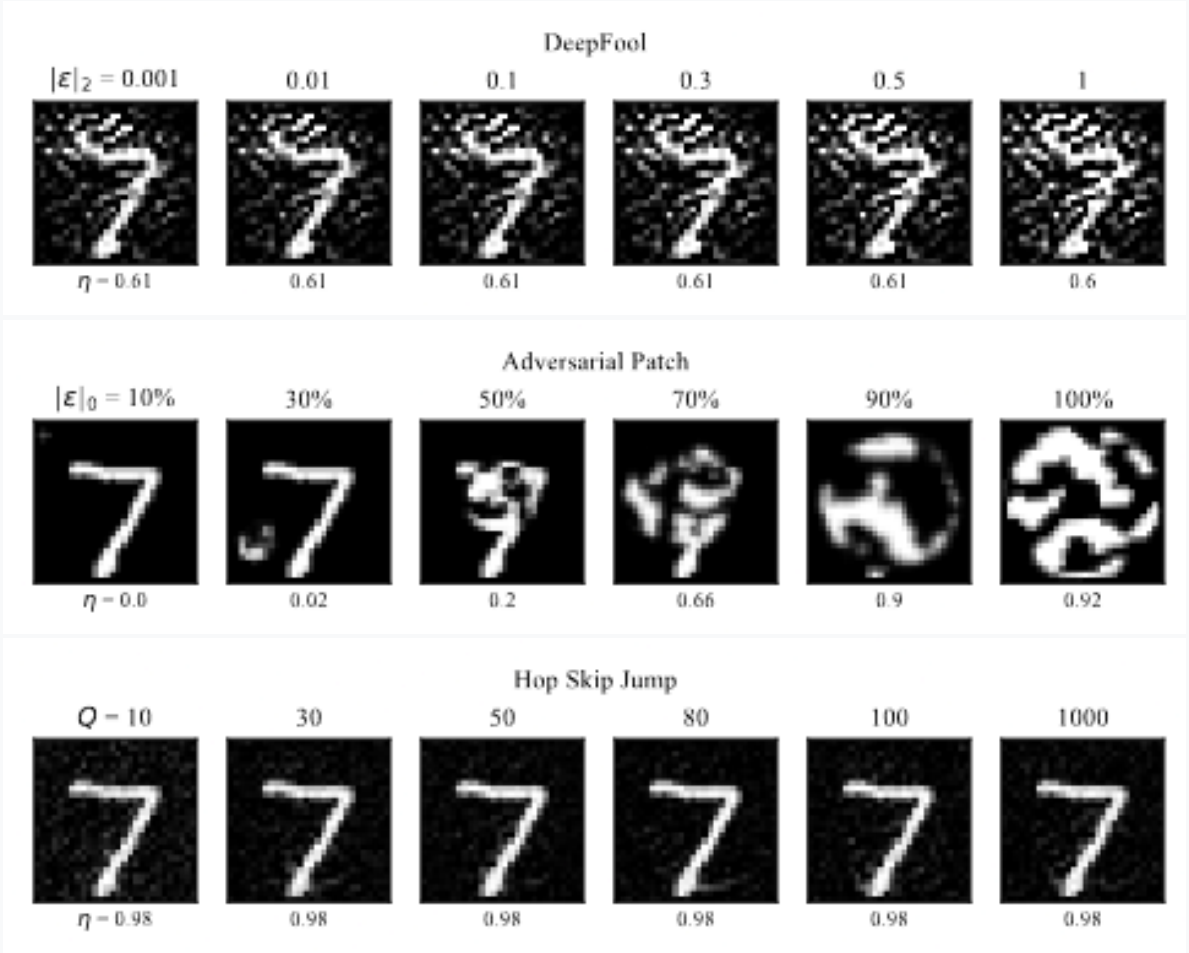
More Attacks

Deep (ℓ_2): the 'Deepfool Attack' (8) finds the minimal separating hyperplane between two classes and then adds a specified amount of perturbation to ensure it crosses the boundary by using an approximation of the model gradient.

Pixel (ℓ_0): the 'PixelAttack' uses a well-known multi-objective search algorithm (9), but tries to maximise false confidence while minimising the number of perturbed pixels.

HSJ (ℓ_∞): the 'Hop Skip Jump' attack is a black-box attack that works on any model. It approximates class boundaries using an offline approximation at the cost of extra computation compared to the above.

Evasion Attacks Visualized



This figure depicts images as we increase the attack strength (ϵ), as well as the adversarial success rate, η.

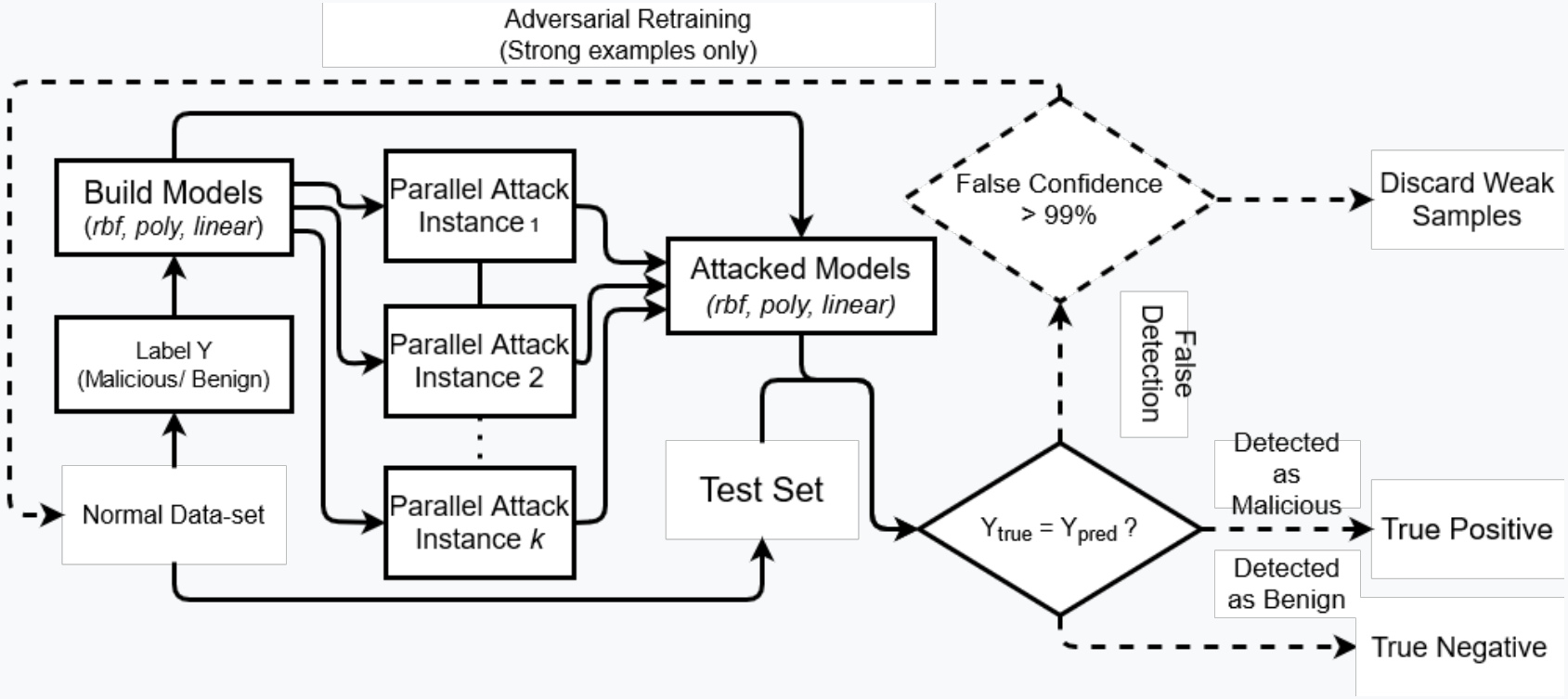


Figure: This diagram illustrates the iterative procedure associated with "adversarial retraining" wherein misclassified adversarial are appended to the training database and the model is retrained.

Lowest-Hanging Fruit: SVCs

Attack Parameters

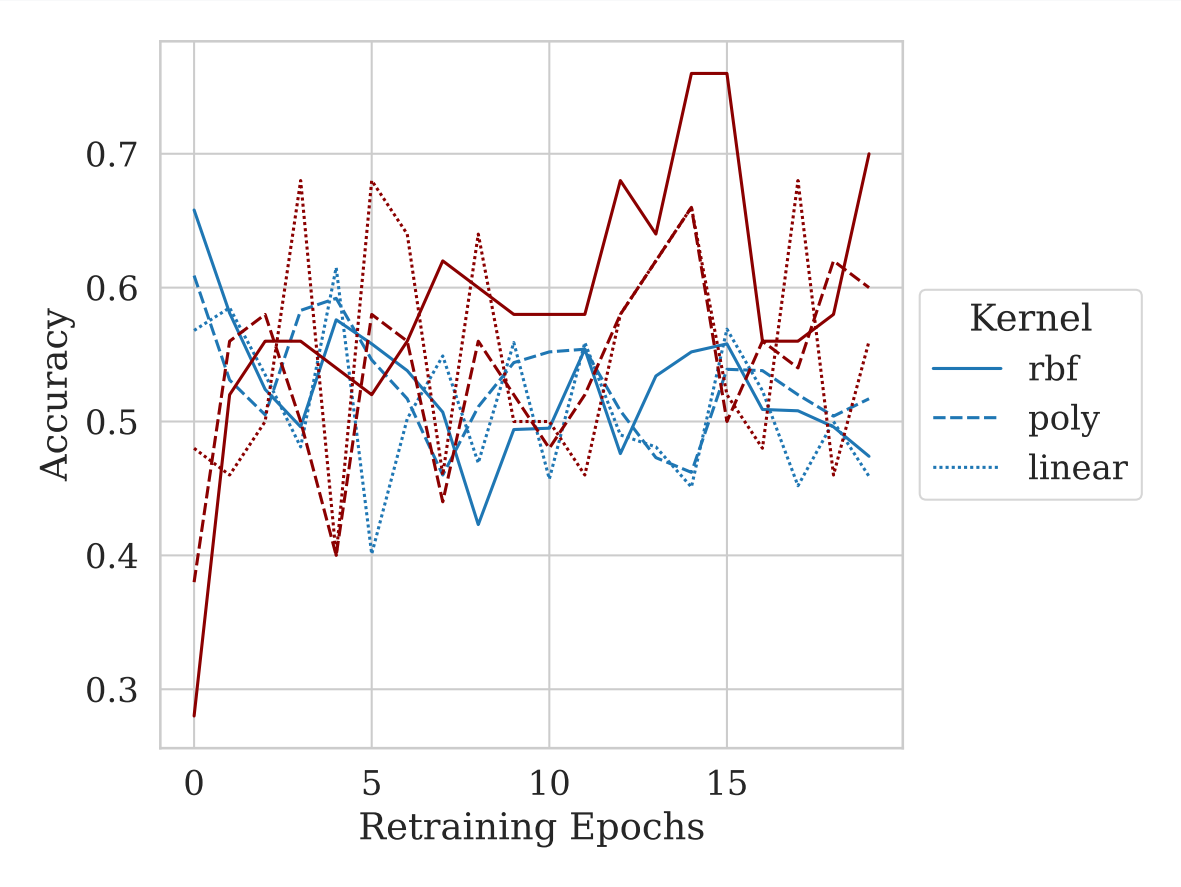
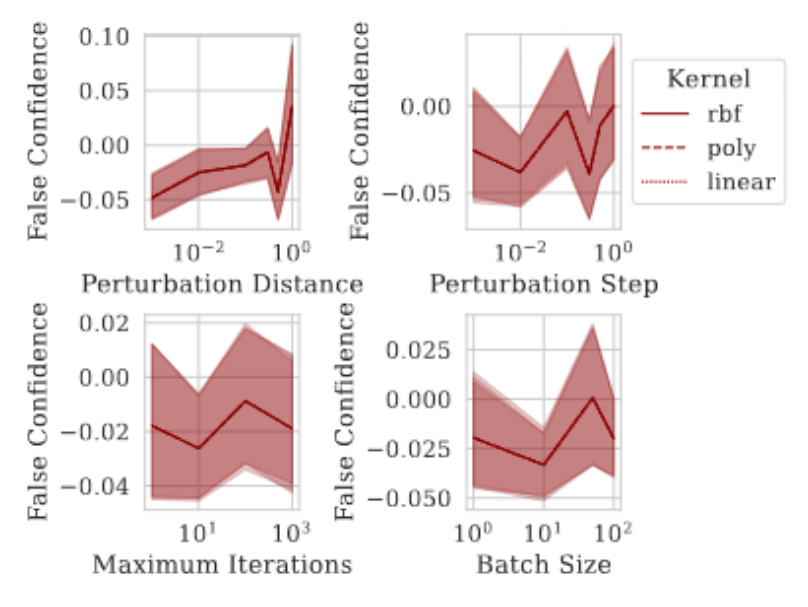
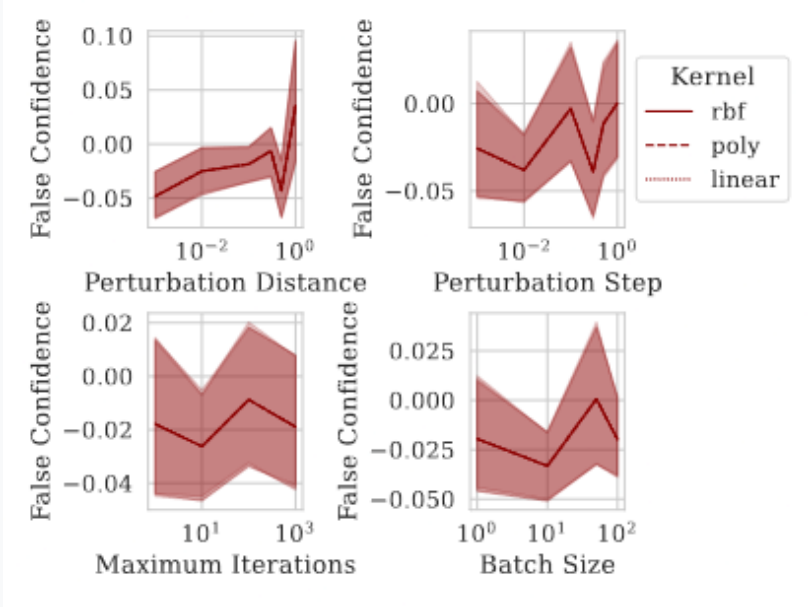


Figure: The accuracy of various models after retraining. We see that retraining improves the adversarial accuracy (red) while hurting the benign accuracy (blue).

Lowest-Hanging Fruit: SVCs

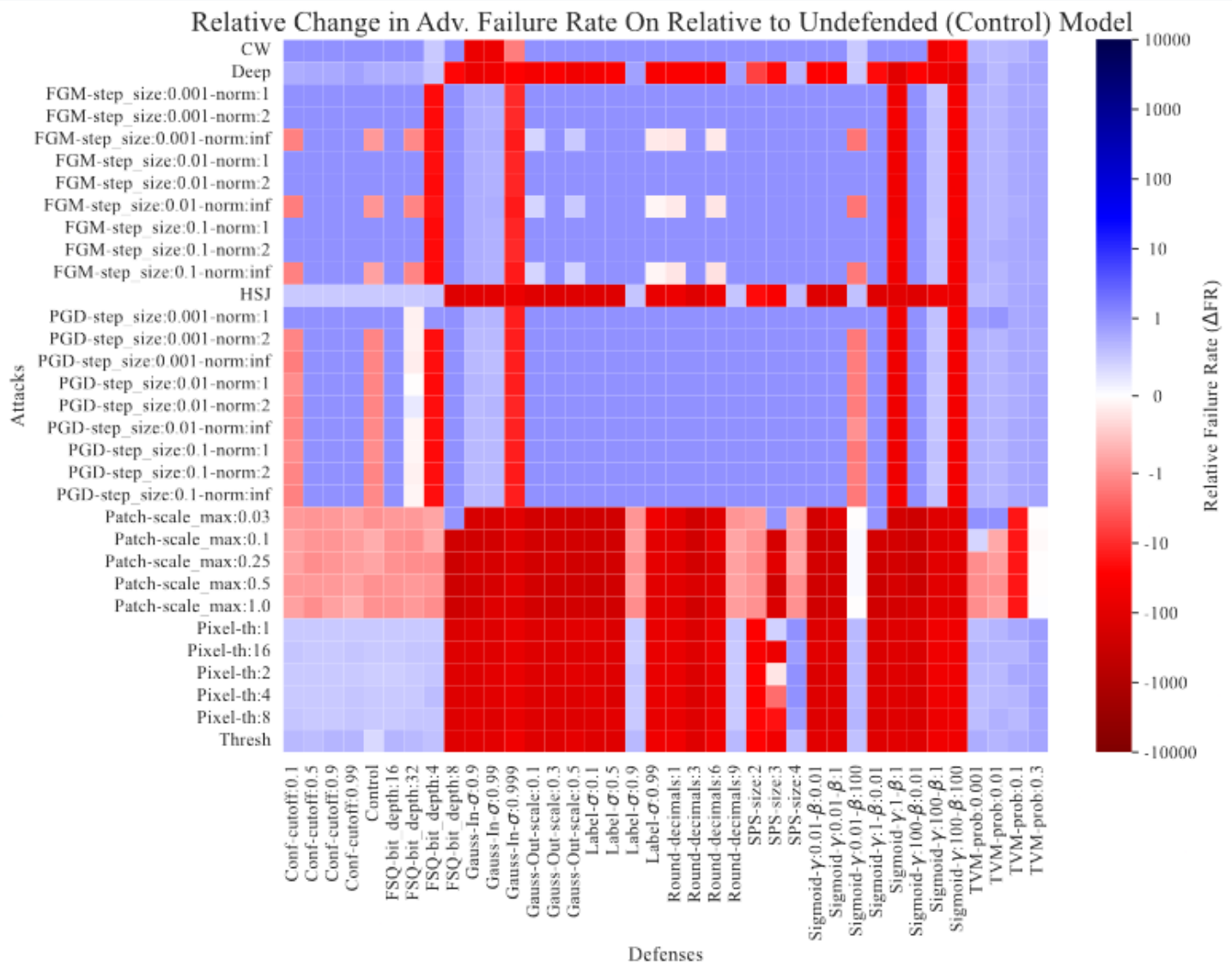
Retraining





We can see that despite improvements in the adversarial accuracy, it does not change the total amount of false confidence aross all samples. That is, improving adversarial accuracy comes at a cost elsewhere.

Computer Vision A Survey



Computer Vision

A Survey (caption)

The figure on the previous slide depicts the relative change in error rate compared to an undefended model for various defences (y-axis) and attacks (x-axis). This clearly demonstrates that improvements against an adversary are inconsistent, at best.

Computer Vision

Deep Problems

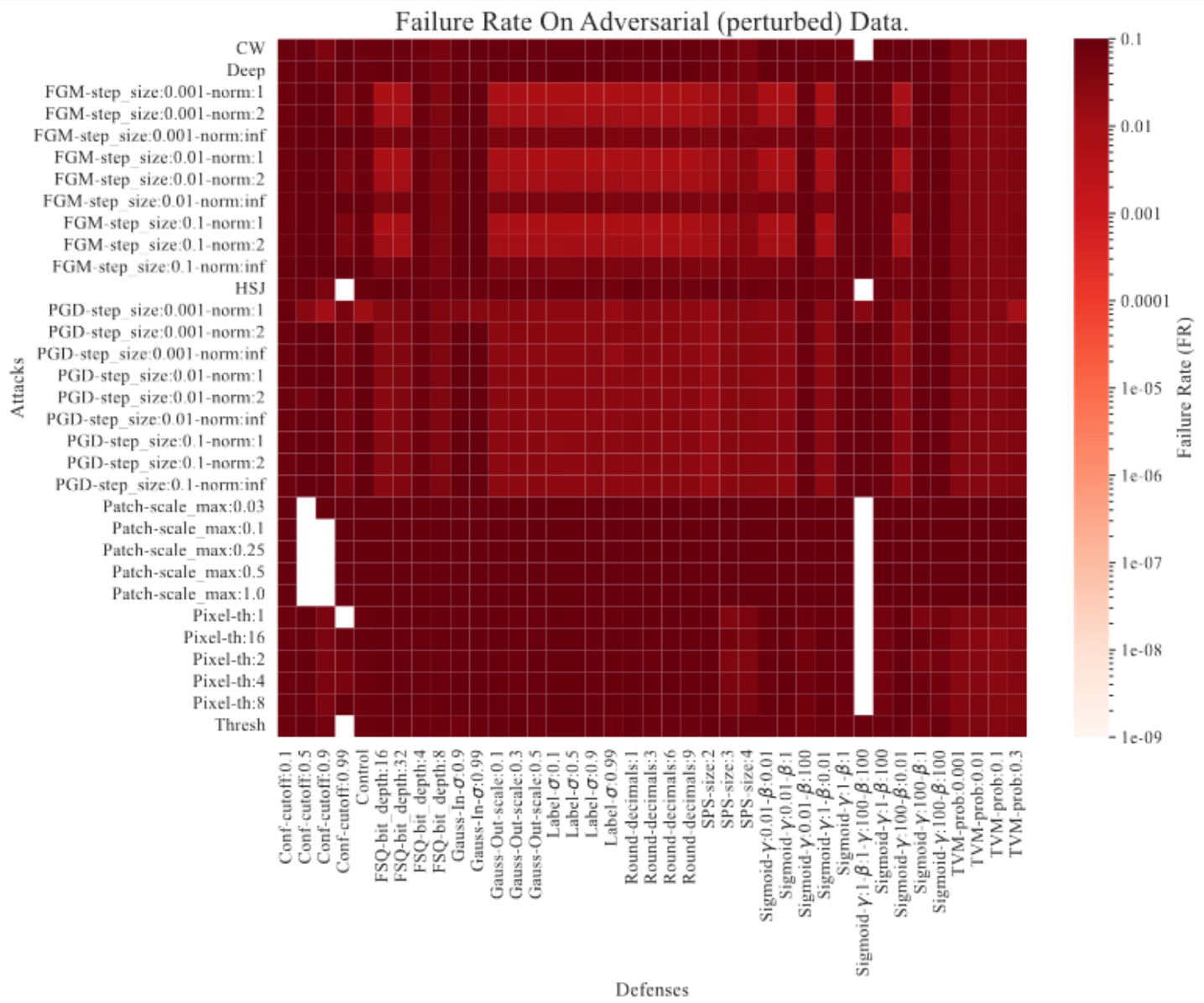
The figure on the next slide depicts the absolute failure rate of each defense and attack combination.

Dark blue (not pictured) represents models that meet legal requirements. Dark red indicates catastrophically dangerous systems.

White indicates that no adversarial examples were found, though this is likely due to computational constraints and not a model's invulnerability to a particular attack.

Computer Vision

Deep Problems



Modelling Failures

Accelerated Failure Models

We can then model the relationship between various model parameters (*e.g.* a defence) and the expected attack time, which is also called *survival time*. Often, it is convenient to assume a relationship between the baseline survival function (S_0) resulting survival time (S_θ) and some acceleration factor, ϕ.

$$S_\theta(t) = \phi(\theta, x) S_0(t) \tag{5}$$

where ϕ is an acceleration factor that depends on the covariates, typically $\phi(\theta, x) = \exp(\theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_p x_p)$.

"Smokers are X times as likely to die prematurely. Coal power generation increases asthma rates by y% compared the baseline."

Modelling Failures

Survival Analysis

We can use the survival function (and some calculus) to compute the average survival time, such that:

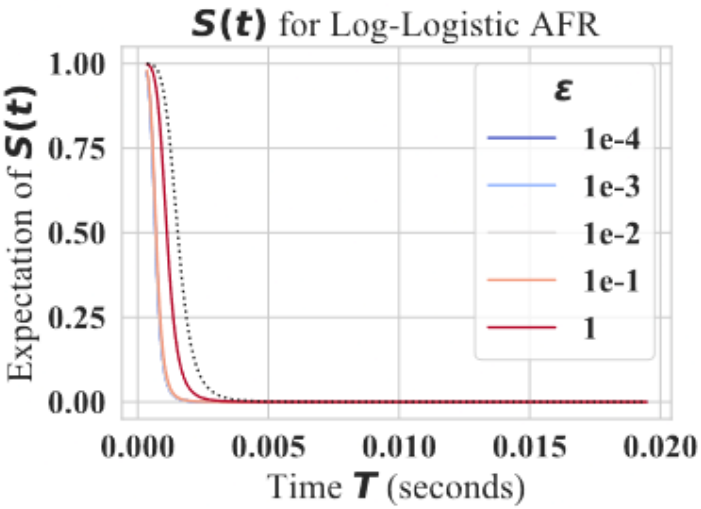
$$\overline{S_\theta} = \int_0^\infty t \cdot S_\theta(t)\, dt, \tag{6}$$

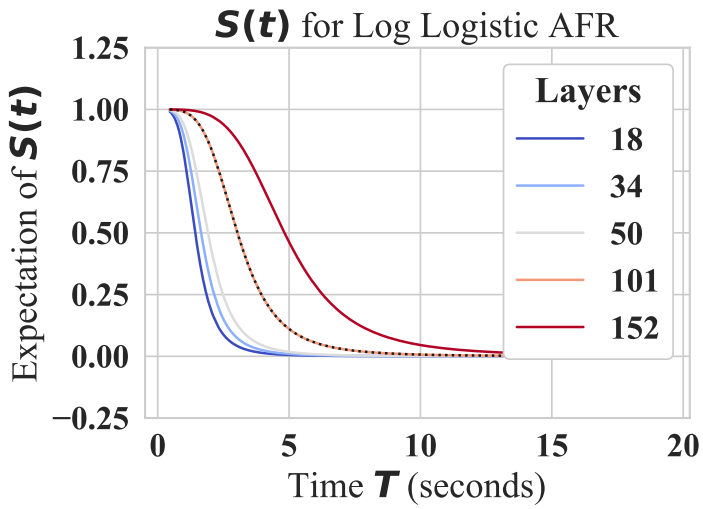
which is measure in seconds. We can then compute the relative to cost to a model builder (t_{train}) to an attacker ($\overline{S_\theta}[T]$, t_{attack})

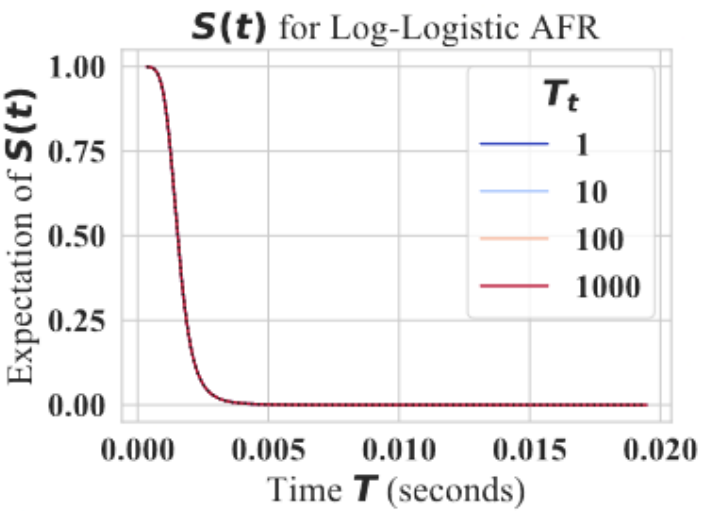
$$\overline{C} = \frac{t_{train}}{\overline{S_\theta}} \approx \frac{t_{train}}{t_{attack}}. \tag{7}$$

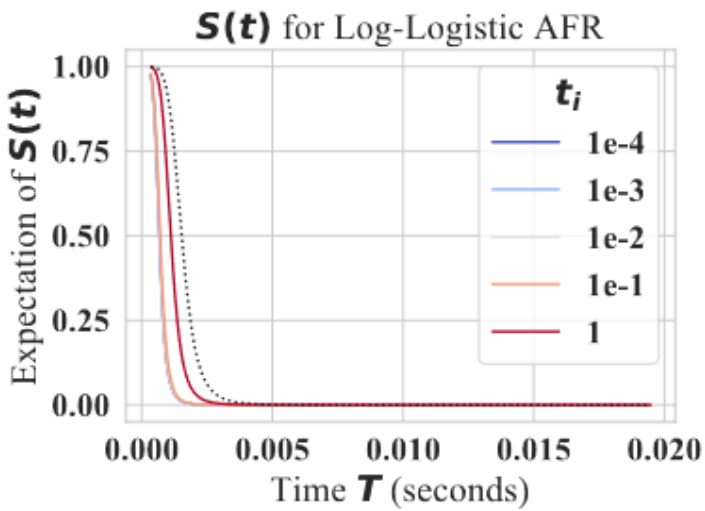
Modelling Failures

Survival Analysis









Modelling Failures

Survival Analysis

The previous slide is captioned here.

Top Left: This plot shows how the survival time (e.g. time to a successful attack) changes as we vary the attack strength, ϵ. It demonstrates a non-monotonic relationship between strength and failure rate, that is measured in partial seconds.

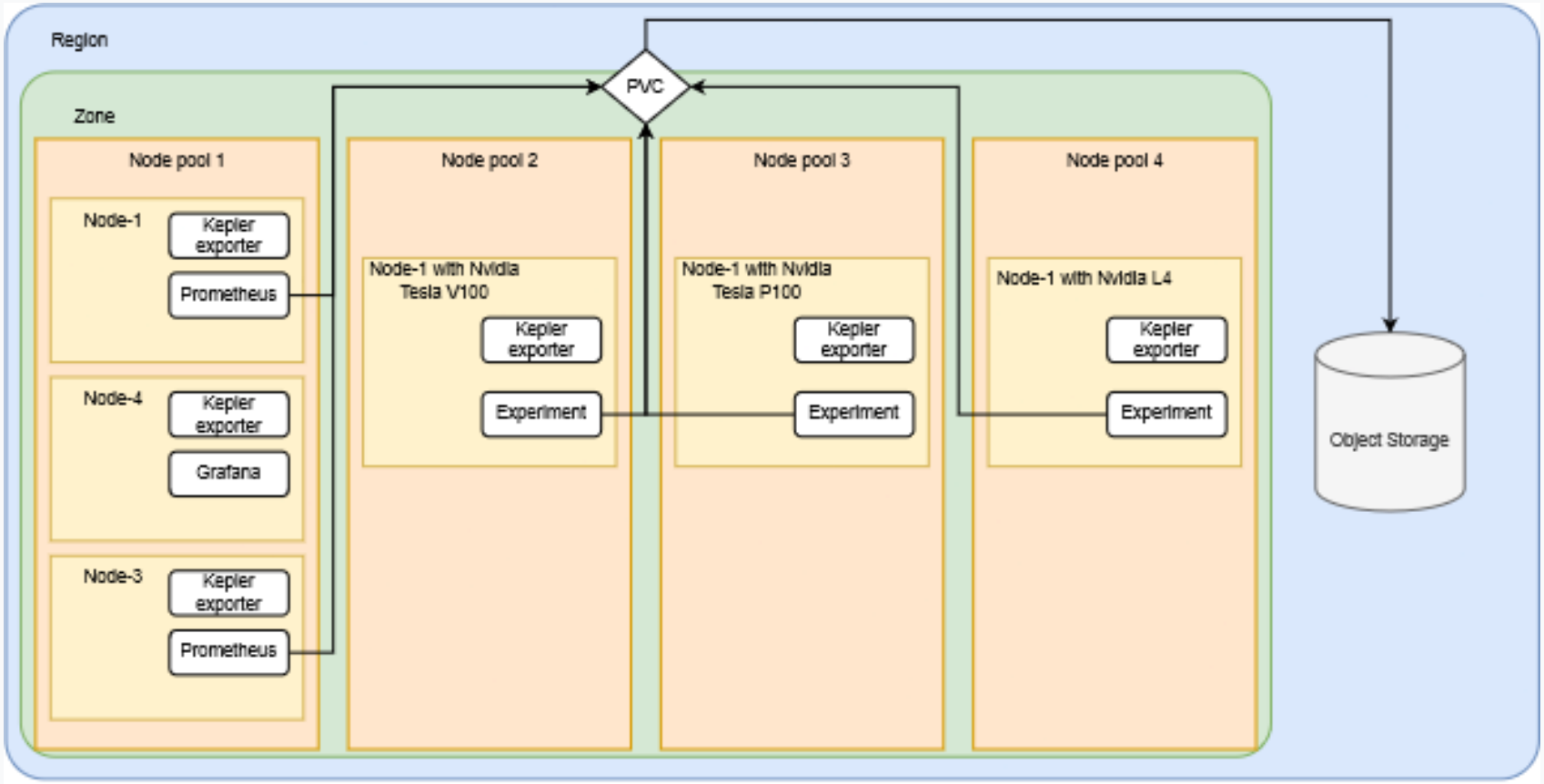
Top Right: This plot shows how survival time changes with the number of hidden layers in ResNet. It shows that deeper models are more robust—to a degree measured in seconds.

Bottom Left: This plot shows how the training time (T_t) changes the survival time $S(t)$. It clearly demonstrates no relationship between these two values.

Bottom Right: This plot shows how the inference time per sample (t_i) changes the survival time $S(t)$. It clearly demonstrates that slower models have lower failure rates.

Modelling Failures

Scaling Up



Modelling Failures

Scaling Up

The previous slide is captioned here.

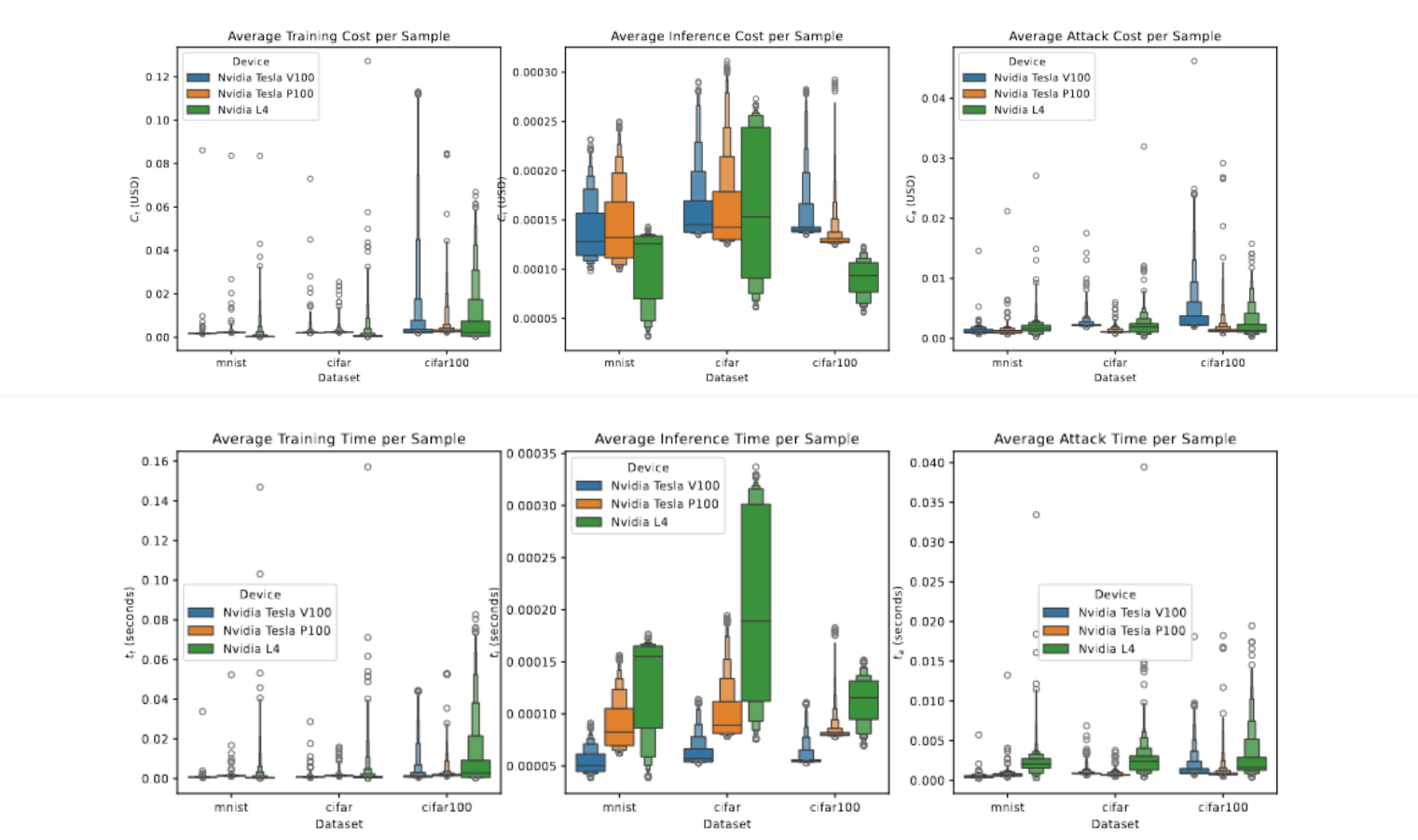
This figure depicts a cloud architecture diagram for the experiments conducted in the following slides. It shows 4 node pools. 1 for each of the tested GPUs and 1 for a monitoring stack that includes KEPLER (power measurement), Grafana (system statistics) and Prometheus (time value database), as well as a persistent volume claim that allows us to mount the object storage asset as a folder inside our node pools.

	V100	P100	L4
Cost (USD/hour)	2.55	1.60	.81
Power (Watts)	250	250	72
Memory Bandwidth (GB/s)	900	732	300

Table: Hardware specifications for tested GPUs. All specifications were retrieved from the Nvidia website at the following links: V100 Datasheet, P100 Datasheet, and L4 Datasheet. Prices were retrieved from Google Cloud Platform for the `europe-west4` region on 3 December 2023 at this link.

Cost

Attacks vs Training



These figures depict the times (top row) and costs (bottom row) associated with training (left), prediction (middle), and attacking a given model on the specified hardware (colour). It is clear that generating a successful attack is much, much easier than training a model when the scales of the state-of-the-art are considered, though attacks are still more costly than inference.

Conclusions

- ResNet is not safety critical. Neither are support vector machines.
- For ResNet18, a successful attack costs as much as training a single sample, but the attacker gains the advantage as we add more layers
- Training time, by itself, has no effect on the robustness
- Inference time and noise distance do change the model robustness
- Bigger models aren't more robust—they're just slower.

[1] International Standards Organization, "ISO 26262-1:2011, road vehicles — functional safety." https://www.iso.org/standard/43464.html (visited 2022-04-20), 2018.

[2] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," pp. 39–57, 2017.

[3] B. Biggio, B. Nelson, and P. Laskov, "Poisoning Attacks against Support Vector Machines," *arXiv:1206.6389 [cs, stat]*, Mar. 2013.

[4] T. Orekondy, B. Schiele, and M. Fritz, "Knockoff nets: Stealing functionality of black-box models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4954–4963, 2019.

[5] Z. Li and Y. Zhang, "Membership leakage in label-only exposures," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pp. 880–895, 2021.

[6] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana, "Certified robustness to adversarial examples with differential privacy," in *2019 IEEE Symposium on Security and Privacy (SP)*, pp. 656–672, 2019.

[7] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," *arXiv:1706.06083*, 2017.

[8] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2574–2582, 2016.

[9] S. Kotyan and D. V. Vargas, "Adversarial robustness assessment: Why in evaluation both ℓ_0 and ℓ_∞ attacks are necessary," *PloS one*, vol. 17, no. 4, p. e0265723, 2022.

Thank you!

https://github.com/simplymathematics/deckard